1993

The organizational preparation of existing relational databases for the integration of expert systems.

Snow, Warren W.

Monterey, California. Naval Postgraduate School;

lassified
rity Classification of this page

REPORT DOCUMENTATION PAGE

Report Security Classification: Unclassified	1b Restrictive Markings
Security Classification Authority	3 Distribution/Availability of Report Approved for public release; distribution is unlimited.
Declassification/Downgrading Schedule	
erforming Organization Report Number(s)	5 Monitoring Organization Report Number(s)

Name of Performing Organization val Postgraduate School	6b Office Symbol *(if applicable)* 39	7a Name of Monitoring Organization Naval Postgraduate School
Address *(city, state, and ZIP code)* nterey CA 93943-5000		7b Address *(city, state, and ZIP code)* Monterey CA 93943-5000
Name of Funding/Sponsoring Organization	6b Office Symbol *(if applicable)*	9 Procurement Instrument Identification Number
lress *(city, state, and ZIP code)*		10 Source of Funding Numbers

Program Element No	Project No	Task No	Work Unit Accession No

Title *(include security classification)* THE ORGANIZATIONAL PREPARATION OF EXISTING RELATIONAL DATABASES R THE INTEGRATION OF EXPERT SYSTEMS (Unclassified)

Personal Author(s) Snow, Warren Wilson

Type of Report	13b Time Covered	14 Date of Report *(year, month, day)*	15 Page Count
ster's Thesis	From To	93, 05, 28	74

Supplementary Notation The views expressed in this thesis are those of the author and do not reflect the official policy or position the Department of Defense or the U.S. Government.

Cosati Codes			18 Subject Terms *(continue on reverse if necessary and identify by block number)*
d	Group	Subgroup	Relational databases, expert systems, artificial intelligence, distributed databases, X.500

Abstract *(continue on reverse if necessary and identify by block number)*

s thesis is a management guide for strategically planning a future integration of relational databases and expert systems. It ites best to an organization with large established relational database(s), that is trying to access the changes required to integrate ert systems with those databases. Technical considerations for such a change are discussed, and include the role of database malization and the requirement to maintain applications that are independent of the database structure. The organizational siderations of such an integration are examined, and focus on the people skills required within an organization to develop and ntain database and expert system combinations. Three product categories are established to represent an integrated system, and ommercial off the shelf product from each category is reviewed to illustrate its specific capabilities. The combination of tional databases and expert systems has the potential to deliver information systems of future strategic importance. This thesis ves to assist the information systems management of military organizations in planning the transition to such a system.

Distribution/Availability of Abstract unclassified/unlimited __ same as report __ DTIC users	21 Abstract Security Classification Unclassified	
Name of Responsible Individual C. Boger	22b Telephone *(include Area Code)* (408) 656-2607	22c Office Symbol AS/BO

FORM 1473,84 MAR — 83 APR edition may be used until exhausted — security classification of this page

All other editions are obsolete — Unclassified

Approved for public release; distribution is unlimited.

The Organizational Preparation of Existing Relational Databases For The Integration of Expert Systems

by

Warren W. Snow
Major, U.S. Air Force
B.S., United States Air Force Academy

Submitted in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE IN SYSTEMS TECHNOLOGY

from the

NAVAL POSTGRADUATE SCHOOL
June 1993

ABSTRACT

This thesis is a management guide for strategically planning a future integration of relational databases and expert systems. It relates best to an organization with large established relational database(s), that is trying to assess the changes required to integrate expert systems with those databases. Technical considerations for such a change are discussed, and include the role of database normalization and the requirement to maintain applications that are independent of the database structure. The organizational considerations of such an integration are examined, and focus on the people skills required within an organization to develop and maintain database and expert system combinations. Three product categories are established to represent an integrated system, and a commercial off the shelf product from each category is reviewed to illustrate its specific capabilities. The combination of relational databases and expert systems has the potential to deliver information systems of future strategic importance. This thesis serves to assist the information systems management of military organizations in planning the transition to such a system.

TABLE OF CONTENTS

I. INTRODUCTION 1

A. FOREWORD 1

B. BACKGROUND 2

C. SCOPE 3

1. Are Structural Changes to Relational Databases Necessary? 3

2. Are Organizational Changes Necessary? 4

D. WHY IS THIS IMPORTANT? 4

II. TECHNICAL ASPECTS OF USING RELATIONAL DATABASES WITH EXPERT SYSTEMS 7

A. OVERVIEW 7

B. EXPERT SYSTEMS PRIMER 7

C. USING RELATIONAL DATABASES WITH EXPERT SYSTEMS . 8

1. Guidance for Accessing Relational Databases from Expert Systems 8

a. Application-independent design for databases 9

b. Loose Coupling of Applications and Data 9

2. Technical Interaction between Expert Systems and Relational Databases 10

a. Structured Query Language (SQL) 11

b. Database Normalization 12

3. To Where Does Relational Database and Expert System Interaction Lead? 13

D. CHOICES IN EXPERT SYSTEM ACCESS TO RELATIONAL DATABASES 14

1. Primer on Client/Server Architecture 14

2. Relational Database Interoperability Architectures 15

a. Database Connectivity Software 15

(1) Advantages: 17

(2) Disadvantages: 17

(3) Future Prospects: 18

b. RDBMS's With Conventional Gateways 18

(1) Advantages: 19

(2) Disadvantages: 19

(3) Future Prospects: 20

c. Open Gateways 20

(1) Advantages: 21

(2) Disadvantages: 21

(3) Future Prospects 22

d. PC Front Ends with Database Application Program Interfaces 22
(1) Advantages: 23
(2) Disadvantages: 24
(3) Future Prospects: 24
e. Database Application Program Interface Alternatives 25
(1) SQL Access Group (SAG) 25
(2) Open Database Connectivity (ODBC) 25
(3) Integrated Database API (IDAPI) 26
E. Repositories 27
1. Repository Theory 28
2. The X.500 Directory Services Standard 29
3. Why are Repositories and X.500 Important? 31

III. ORGANIZATIONAL IMPACT ON DATABASE MANAGEMENT FROM EXPERT SYSTEMS 32
A. OVERVIEW 32
B. THE PEOPLE ROLE IN MANAGING RDBMSs 33
1. Data Administrator 33
2. Database Administrators 35
3. Upper Management 37
C. THE PEOPLE ROLE IN MANAGING EXPERT SYSTEMS 38

1. The 'Expert' 38

2. Knowledge Engineer 39

D. NEW ROLES FOR COMBINED OPERATIONS 40

1. Knowledge Administrator 41

2. KnowledgeBase Administrator 42

a. Using Expert Systems to Manage Repositories 44

3. Upper Management 45

IV. COMMERCIAL PRODUCTS - THEIR POTENTIAL FOR COMBINED USE 46

A. OVERVIEW 46

B. PRODUCT CATEGORIES 47

1. Expert Systems - Nexpert 48

2. Middleware - SequeLink 50

3. Relational Database Management System - Sybase 51

a. Sybase Open Client 52

b. Sybase Open Server 52

c. Sybase Open Gateways 53

d. Sybase Database Remote Procedure Calls (RPCs) 54

e. Sybase Summary 55

V. CONCLUSION 57

LIST OF REFERENCES 59

INITIAL DISTRIBUTION LIST 63

LIST OF FIGURES

Figure 1: Database Connectivity Software 16

Figure 2: RDBMS with Conventional Gateway 19

Figure 3: Open Gateways 21

Figure 4: Application Program Interfaces 23

Figure 5: X.500 Query Process 30

Figure 6: Product Categories 47

Figure 7: Stored Procedures 55

I. INTRODUCTION

> All economic systems sit upon a 'knowledge base.' All business enterprises depend on the preexistence of this socially constructed resource. Unlike capital, labor, and land, it is usually neglected by economists and business executives when calculating the 'inputs' needed for production. Yet this resource -partly paid for, partly exploited free of charge- is now the most important of all. (Toffler, 1990)

A. FOREWORD

In his book PowerShift, futurist Alvin Toffler describes a 21st century dominated *not* by wealth or violence (as in the past), but by knowledge. He predicts knowledge will become the predominant source of power, if it has not already (Toffler, 1990). Current management literature is replete with references to the rapid growth of knowledge, and the ramifications of managing this growth (or of failing to do so). In his latest book Liberation Management, Tom Peters (author of the classic In Search of Excellence) devotes a significant portion of his 800-page management guide to the topic of knowledge management (Peters, 1992). In example after example he illustrates how tomorrow's most successful companies will be those organized to make the best use of their peoples' skills, and able to use technology to manage the knowledge that exists within their companies today. An appraisal of these books, and other ones, reveals some major recurring themes. Foremost is the significance of ongoing rapid growth in information technology. Second is the growing value of knowledge as a tangible commodity, much like we have placed tangible value on capital, labor, or land in the past. As we enter what Toffler, and many others, call the Information Age, an organization's ability to use

its people and technology to manage knowledge will be instrumental to its ability to compete. Two technology ingredients of the Information Age are relational databases and expert systems. As relational database technology evolves, and expert systems begin to mature into widespread use, the effects of integrating these two technologies offer the potential for synergistic benefits far beyond the advantages of focusing on each technology alone. This thesis will explore some of the technical and organizational ramifications we can expect, and how to deal with them, as the evolution of these technologies continues.

B. BACKGROUND

This thesis is a management guide for future strategic planning for relational database systems as they relate to expert systems. The reader is assumed to have a general understanding of relational databases and expert systems. This study relates to an organization having a large established relational database(s) and contemplating a move toward using expert systems in conjunction with their established databases. The purpose of assuming that existing relational databases are in use (vice older technology such as hierarchical database systems) is as a means of limiting the scope of this thesis, and to more precisely target its information to the organizations that are most likely to need it. The organization's particular hardware architecture is not a critical factor to this study if the relational databases are accessible via Structured Query Language queries. In the cases where it's necessary to specify the hardware architecture, client/server configurations will be used (i.e., relational databases residing on servers accessible by

applications residing on clients). A notional military organization that fits this description is a service-level personnel command. Maintaining the personnel records of all members of a military service is clearly a large-scale database function, and many recurring personnel-oriented activities lend themselves to expert systems. Does the future hold a role for an expert system to assist your promotion board in making fair and unbiased promotion decisions? Would you benefit from your detailer having the assistance of an expert system that recommends specific career options, tailored to your individual needs and the Service, based on all information in today's assignments database? Could a personnel command function more effectively if these expert systems, and others, were in place?

C. SCOPE

The scope of this thesis will consider future strategic planning for relational database systems in the context of two specific questions. This doesn't imply they are the only important questions, just two that are worthy of detailed inspection.

1. Are Structural Changes to Relational Databases Necessary?

When planning for the integration of expert systems to an information system, are structural changes to relational databases necessary, and if so why?

- What kinds of data (i.e., text, image, numerical, video...) can expert systems use, and how does that differ from the contents of relational databases?
- What are the similarities and differences between relational databases and knowledge bases?

- Should a data dictionary change to accommodate the needs of expert systems? Is there a role for a 'knowledge dictionary' when an organization's use of expert systems becomes widespread? If so, what is it?

- Should relational database schemata be adapted to accommodate the needs of expert systems? If so, how should they be changed?

2. Are Organizational Changes Necessary?

Are changes to the organization (i.e., the people who perform data administration and their responsibilities) necessary to have relational databases serve the information needs of expert systems? The thrust of this portion of the thesis is to look at the people implications of using expert systems with relational databases. Among the issues to be covered are:

- Should the functions people perform to maintain relational databases change to accommodate the use of expert systems?

- Should people performing traditional database functions (i.e., database administrator) gain counterparts (i.e., knowledge administrator, knowledge-base administrator) when expert systems gain widespread use in an organization?

D. WHY IS THIS IMPORTANT?

To some readers, this topic may seem of minor significance, especially if expert systems do not loom on the horizon as important to their organization's future. Despite that view however, the evidence from leading edge corporations suggests an inevitable trend toward knowledge management as one of the major functions of information systems. As further military budget cuts occur, wiser fund expenditure will be required to accomplish work more effectively, making better decisions, with fewer people. Expert systems offer this potential, especially in information-laden environments where

smarter decisions can be made more effectively if voluminous amounts of information can be brought to bear on the problem.

As expert systems technology continues to improve, it will reach the potential for widespread use. Unfortunately, the niche expert systems have developed is that they work best in narrow problem domains. This results in expert systems tending to be standalone programs that solve specific narrow problems that are not integrated into the bigger information systems picture. Expert systems do not have to remain in this niche since proper application of database technology can make vast amounts of information available to the power of expert systems, resulting in higher valued knowledge. Access to databases can allow expert systems to become more powerful, provide more timely advice, and most importantly, become strategic information system assets.

Merging relational databases and expert systems technology to manage knowledge can spur a requirement to change information systems organizations. Managing knowledge, instead of data, should force us to pause and re-think the role of database administrators. The addition of new functions, such as knowledge engineers, should be seen as an opportunity to reconsider the traditional roles of all information technology players (programmers, operators...).

Many of today's leading companies are focusing their energy on the challenge of managing knowledge. When done right, their efforts allow them to downsize their mainframe-based information systems into client/server-based architectures, and accomplish tasks more effectively with less, although more highly-skilled, people. The

points outlined above are but a few of the many reasons why this area will continue to grow in importance.

II. TECHNICAL ASPECTS OF USING RELATIONAL DATABASES WITH EXPERT SYSTEMS

A. OVERVIEW

The objective of this chapter is to discuss the technical aspects of using expert system applications with relational databases. It begins with a brief primer on expert systems, and then presents two important concepts in planning information systems where applications access databases. A technical explanation then describes how expert systems access relational databases to obtain information. This leads to the point that making structural changes to relational databases to accomodate the needs of expert systems is not required or desirable. Then four database access architecture choices are outlined and their pros and cons are discussed. Lastly, the future-oriented topic of data repositories is discussed. Repositories encompass several future information system trends; an understanding of them can prove valuable in planning future information systems.

B. EXPERT SYSTEMS PRIMER

Expert systems (ES) are computer-based applications, within the field of artificial intelligence, that use a knowledge base developed from human expertise for problem solving (Freedman, 1992). Once developed, these systems perform a consultation with a human user by asking a series of questions relating to the particular problem it is

designed to solve. The user consultation, as well as the reasoning process within the application, is controlled by the inference engine, which is a major component of ESs. The inference engine processes user-provided information through the knowledge base to derive answers, or provide advice, to the user. The knowledge base is a set of rules developed for use within the ES based on interviews with human experts in the field of interest, or from documented sources of expertise.

C. USING RELATIONAL DATABASES WITH EXPERT SYSTEMS

By using rule-based expert systems with relational databases, the ES gains access to vast sources of information that can assist in the consultation process. In the course of an ES consultation, information available to the ES can come from the user, from within the knowledge base, and from an external data source. External databases can provide valuable and timely information to strengthen applications in powerful ways. Wal-Mart, for example, has an application that accesses national weather databases to decide the optimum timing to stock snow shovels in its stores (Caldwell, 1993, pp. 35). This Wal-Mart application illustrates the advantages to applications that can be gained by regarding information accessibility as a strategic asset.

1. Guidance for Accessing Relational Databases from Expert Systems

There are two primary concepts one should follow when planning future systems in which applications will take advantage of databases.

a. Application-independent design for databases

An application-independent design for databases holds that one should be primarily concerned with the organization of the data itself in a database rather than how the data will be used by an application (Date, 1991, pp. 523). The main reason application-independent design is important is that all future uses for data can't be known at the time of database design. If a database is to retain the ability to become a future strategic asset, then its design must be robust and independent so future application needs will not invalidate the database structure (Date, 1991, pp. 523).

Application-independent design also insulates the information resource from future technology advances. In the same way that all future uses of data can never be known at design time, neither can one know all future technology advances at design time. As expert systems technology matures, making use of those advances should not require changes to the database structures they may access. To develop a database of lasting value, it's vital that the database be of application-independent design.

b. Loose Coupling of Applications and Data

A loose coupling approach suggests that applications and databases should remain distinct, but communicate via a call-based interface between the two (Date, 1991, pp. 671). While a definite 'seam' remains between these components, the call-based interface allows for data query and retrieval between the expert system and the database. A call-based interface implies that the application performs logic operations, and then makes 'calls' to databases to perform database operations and return information to satisfy requests from within the application.

The loose coupling approach is also the basis for providing the flexibility to interface multiple applications to multiple databases in a wide variety of ways. A single application, such as the Wal-Mart example mentioned earlier, may call upon multiple weather databases in different regions to optimize snow shovel stock levels. Conversely, multiple product applications (perhaps snow shovels, umbrellas, and suntan lotion) may call upon one national weather database to help optimize their stock levels. Also, future advances in expert system and SQL technology may some day allow for 'smart' queries that go out and find the best database to provide information to an expert system. In all of these cases, the loose coupling approach keeps the data design separate from the application, and therefore ready to satisfy tomorrow's yet-to-be-determined application requirement.

For relational databases, the Structured Query Language (SQL) is the call-based standard that provides this interface for applications. As will be shown next, SQL provides a standard that is met by all relational database management systems (RDBMS), and is callable by expert systems as well as other types of applications.

2. Technical Interaction between Expert Systems and Relational Databases

With the concepts of application-independence and loose coupling in mind, it's important to have a technical understanding of how expert systems and relational databases interact. The Structured Query Language (SQL) standard and database normalization provide the basis for such an understanding.

a. *Structured Query Language (SQL)*

SQL began in the mid 1970's as an IBM-developed language called SEQUEL that was used to access the relational databases that ran on IBM mainframe computers (Salemi, 1993, pp. 27). The name was later changed to SQL, which has evolved to become the de facto database query language standard. In 1986, the American National Standards Institute (ANSI) formally published the first SQL standard, referred to as SQL86. Three years later, ANSI adopted an upgraded version of the language called SQL89, or commonly referred to as SQL2 (ANSI, 1989, pp. iii). The International Standards Organization also adopted SQL89 as the standard for database query language (Seybold, 1991, pp. 6).

An SQL query begins as code embedded within the program of an application, in our case an expert system. As one might expect, an ANSI standard also exists which defines Embedded-SQL, allowing SQL commands to be placed as-is within programs written in Ada, C, Cobol, Fortran, Pascal, or PL/I (ANSI, 1989, pp. 9). SQL commands that perform queries or database updates make up the Data Manipulation Language (DML) component of SQL (Viescas, 1989, pp. v). The two other components of the language are the Data Definition Language (DDL) and the Data Control Language (DCL). Upon execution, the embedded SQL commands are translated into database procedure calls, and then passed to the specified DBMS for processing. The commands may pass directly to a DBMS on the same computer, or may traverse one or more networks to reach a DBMS on a separate computer. Once passed, the RDBMS executes the SQL command against the data tables it manages. The DBMS may temporarily join

tables together, or perform other manipulations, in order to extract a copy of the requested information which is then returned over the network(s) to the expert system application. The expert system can then use the information as part of its consultation process. To again cite the Wal-Mart example, the expert system might query a national database to obtain the current snow conditions for areas where Wal-Mart stores are located.

The significance of SQL being such an established and recognized standard is that all relational database products accept the full range of standard SQL statements, as well as additional SQL functionality which many vendors provide to entice customers. SQL has recently begun to gain even more industry attention as groups such as the Open Software Foundation, XOpen, and the SQL Access Group have joined in to push for requirements in the next standard, now being referred to as SQL3 (Seybold, 1991, pp. 7). Users and vendors pay close attention to SQL in the standards process since it lies at the crux of so many technologies, and its use is becoming more and more critical to distributed interoperative information systems of the future.

b. Database Normalization

Database normalization is an element of application-independent design. Normalization can be generally defined as a set of procedures for efficiently organizing the information in a database. More specifically, normalization technically defines a series of steps by which a database administrator should separate large data sets into subsets of related tables. Normalized data tables minimize redundancy within a database, and eliminate the possibility of update anomalies that could otherwise occur on non-

normalized data during SQL data modification transactions (Hansen, 1992, pp.184). In short, a normalized database insures the integrity of its data regardless of the SQL functions that may be performed on that data. Normalization allows SQL activities of an independent application to interact with a DBMS without posing a risk to the database.

3. To Where Does Relational Database and Expert System Interaction Lead?

The important point to make from having a technical understanding of how expert systems and relational databases interact is that properly normalized databases do not and should not modify their structures to accommodate the needs of expert systems. When database resources can offer valuable sources of information to expert system applications, those applications should independently make use of those resources by relying on the SQL standard as the means of interacting with databases. With proper database normalization and use of standard SQL, databases can provide flexible accurate response to queries from expert systems. As more databases become available, including an increasing number of public access databases such as the Wal-Mart weather example, the resources exist to provide expert systems with an ever-growing variety of timely, accurate, and detailed information. To modify relational databases so they accommodate the particular needs of a given expert system, or any other application, is to potentially compromise the value of that database to other applications that make use of that data now or at some point in the future.

D. CHOICES IN EXPERT SYSTEM ACCESS TO RELATIONAL DATABASES

Although the fundamentals of normalization and SQL queries are straightforward (and now covered), the variety of choices on how expert systems can access relational databases are constantly changing due to the emergence of new products, standards, and methodologies. These choices become more complicated if an expert system is required to access multiple databases. This section provides a brief primer on client/server architectures, and then discusses four different relational database access architectures, and explains the pros and cons of each.

1. Primer on Client/Server Architecture

Today's application and database systems are commonly based on a client/server architecture. In this set-up, applications reside on PC or workstation computers referred to as clients. Database transactions are initiated from the client application, over a network, to the RDBMS residing on a server computer. The server's hardware platform may be anything from another PC to a large mainframe. The network may be a Local Area Network (LAN), a Wide Area Network (WAN), or a mixture of different networks. Most of the recent change requests to SQL are aimed at further standardizing the accessibility through networks of applications and distributed databases. A distributed database implies that a single application can operate on data that is distributed across multiple DBMSs, running on different hardware platforms under different operating systems, and connected by different networks (Date, 1991, pp. 617). From the client's viewpoint, the distributed database transparently appears as if it were being managed by one RDBMS residing on one server. In a distributed database

environment, the SQL standard is the basis of agreement from which all distributed database component vendors design their products so they can work together to provide client-transparency. However, planning the means by which network access takes place between applications and databases is a complex task. Even within small standalone networks that handle a few applications and one database, making the right decisions over access can provide the future ability to expand the network so applications can interoperate with multiple or distributed databases. Establishing reliable access to distributed database systems poses a large challenge to expert system planners who want their applications to interoperate with databases.

2. Relational Database Interoperability Architectures

The category of products that provide access from client-applications to server-databases is generally referred to as middleware (Finkelstein, 1993, pp. 46). Middleware products are numerous, and many are narrowly designed to provide specific connectivity between particular components for niche markets. The sheer number of middleware products adds a degree of confusion to this area that can be somewhat resolved by understanding the general architectures for relational database interoperability. Here are four such architectures and their associated advantages and disadvantages (Rymer, 1992, pp. 8).

a. Database Connectivity Software

Database connectivity software products serve to route SQL queries from client applications to server RDBMSs over networks that may contain multiple protocols

(Rymer, 1992, pp. 11). As shown in Figure 1, the connectivity software resides on both the client and server hardware platforms, and is configured to translate among multiple

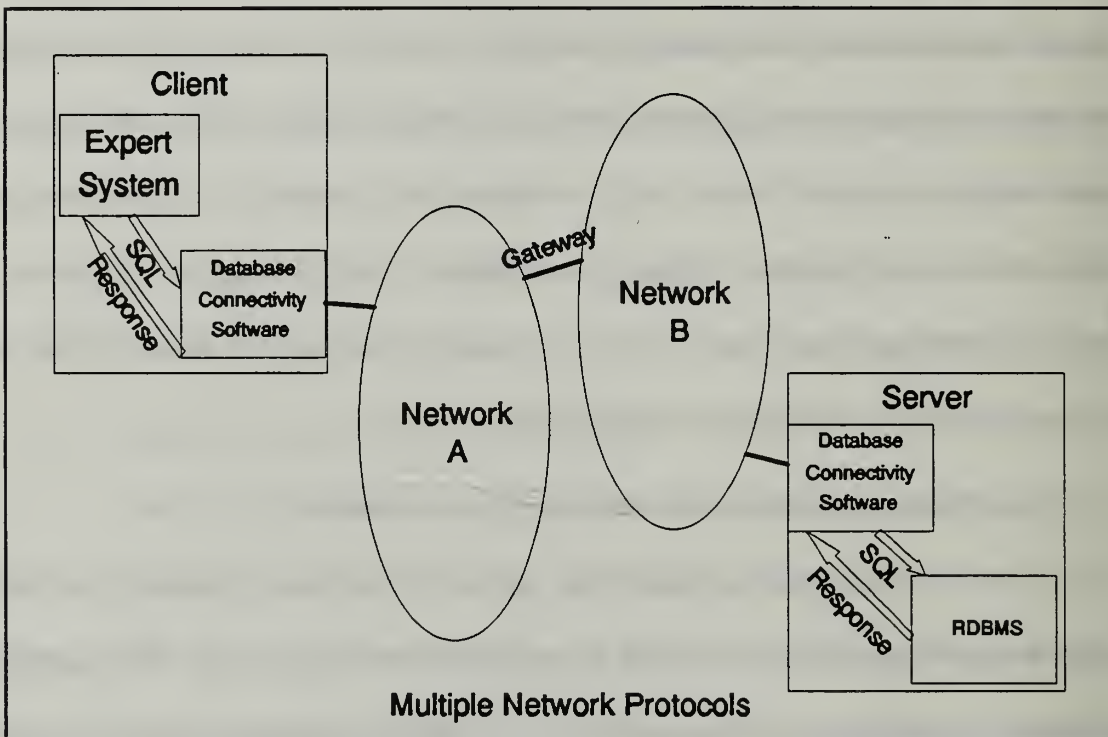

Figure 1: Database Connectivity Software

network protocols to deliver the query to the targeted database, and return the response to the client application. A typical situation that might call for this type of solution would be a network of client workstations tied to a LAN (Network A), which is in turn gatewayed to an IBM mainframe with its own network (Network B). The differing protocols between the LAN and the IBM nework would be negotiated by the database connectivity software residing on the workstation and the mainframe.

The gateway that connects the two networks serves to convert differing protocols between the networks (Finkelstein, 1993, pp. 49). In Figure 1, for example,

Network A might represent a Local Area Network (LAN) using TCP/IP as its network protocol. Network B represents a Wide Area Network (WAN) to a remote mainframe file server using IBM's LU6.2 network protocol. Software in the gateway converts between the two protocols so the query and response can pass between the connectivity software modules transparently.

(1) Advantages: Database connectivity software products tend to be specialized to the particular client, RDBMS, and network protocols the customer has in use. For organizations with existing networks of unusual combinations, database connectivity software may offer the only alternative for database access (Rymer, 1992, pp. 11).

These products work well when the interoperability requirement between an application and a database is limited to specific systems, and is unlikely to grow over time.

(2) Disadvantages: Current database connectivity software is limited in its ability to allow single queries to operate on multiple databases. It usually allows one client to access a single RDBMS (Rymer, 1992, pp. 11). If an expert system required access to multiple databases, it would have to be accomplished by sending a separate SQL query to each RDBMS, receive and combine the responses and then execute further processing within the expert system to consolidate the information for use within the expert system.

Due to their specialized nature, these database connectivity products tend to lack the flexibility to accommodate configuration changes to network protocols, client applications, or server databases (Rymer, 1992, pp. 11).

An organization that depends on this solution for access to multiple heterogeneous databases can soon find themselves mired in the maintenance of a 'spaghetti' network of single links between applications and databases.

(3) Future Prospects: Database connectivity software products will continue to fill the specific need to connect applications to databases through particular combinations of network protocols. However, as organizations continue the trend to downsize mainframe databases onto server platforms, the number of older mainframe-controlled networks will diminish, and the requirement to pass queries over unusual combinations of network protocols will be reduced. As a result, the need for database connectivity software products is likely to diminish.

b. RDBMS's With Conventional Gateways

This method of accessing multiple databases uses a middle tier RDBMS to act as an intermediary to multiple database sources (Rymer, 1992, pp. 12). As illustrated in Figure 2, the intermediary database is linked to multiple databases via gateways. To a client application, the middle tier RDBMS appears as one consistent data directory access structure that responds to all queries. In fact, the middle tier RDBMS accepts queries from applications, compares the query against its 'catalog' of remote databases, and routs the query to the relevant RDBMS. This RDBMS to RDBMS

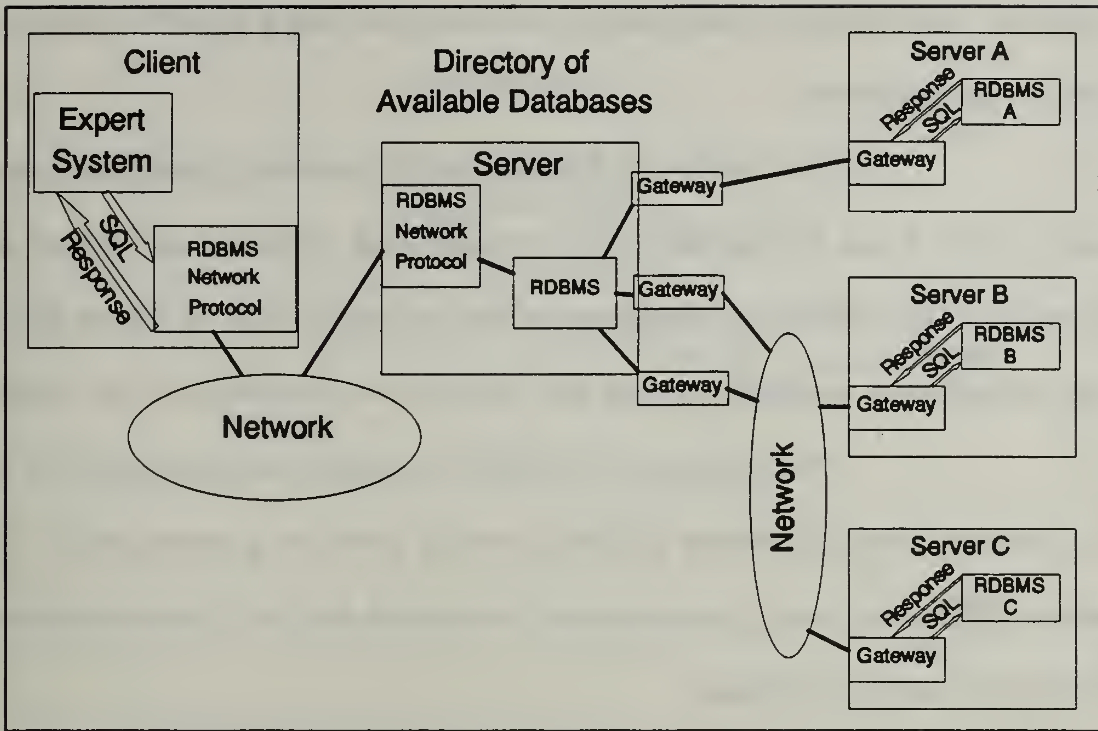

Figure 2: RDBMS with Conventional Gateway

interaction takes place via gateways that are able to accommodate differing network protocols and/or unique add-on SQL features of the distant-end RDBMS. A reverse trip is made to return the results of the query to the original application.

(1) Advantages: Providing data access via a middle tier RDBMS provides a stable and transparent environment to the application programmer for multi-database access (Rymer, 1992, pp. 12). An expert system developer would need to know only one access method make use of multiple databases of potentially varying standards.

(2) Disadvantages: While simplifying the life of the front end developer, the middle tier database is a duplication of data definitions in the distant-end

databases. Maintaining this duplication is both costly and adds a layer of configuration management complexity.

The middle tier RDBMS, and its associated gateways, becomes crucial in that it can become the limiting factor on what other database products are accessible. If the middle tier RDBMS vendor does not support access to a given product (i.e., no gateway is available) then that data source is not accessible with this method.

The selection of the middle tier vendor locks the organization into that vendor's family of products (RDBMS, network protocols, gateways, etc.). This selection becomes an overly critical decision to the future direction of the organization's information systems architecture.

(3) Future Prospects: Although the vendors who offer RDBMSs with conventional gateways are scrambling to offer a wider array of sophisticated services, the future growth of this solution is unlikely (Rymer, 1992, pp. 14). Using this approach is more costly, maintenance intensive, and ties an organization too closely to a non-open solution that's overly dependent on one vendor's family of products.

c. Open Gateways

The open gateways (Figure 3) approach is similar to conventional gateways approach mentioned previously. Open gateways allow for the same transparent connectivity between a client's application and a server's database as with conventional gateways, without the need for an intervening RDBMS to interpret queries and route them to the proper database. Open gateways are also commonly referred to as Universal

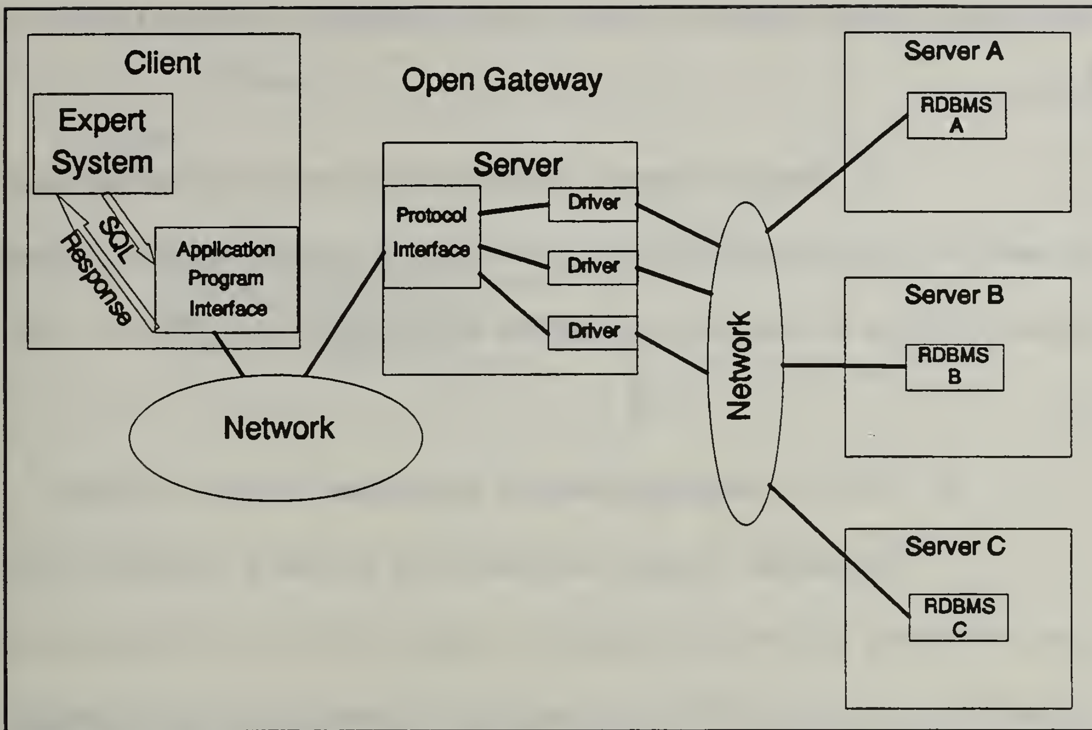

Figure 3: Open Gateways

Gateways (Radding, 1993, pp. 33). An example of an open gateway is Information Builder's EDA/SQL product. EDA/SQL provides access to 50 different RDBMSs which could reside on 35 different platforms (Radding, 1993, pp. 33).

(1) Advantages: Open gateways are more flexible than conventional gateways because they tend to handle more DBMS products and distant end hardware platforms.

The maintenance and configuration management workload of an open gateway is much lower than that of a conventional gateway.

(2) Disadvantages: Open gateway products are still maturing. As a result, different vendor's offerings vary widely in their sets of features. For example,

some products in this category are limited to read-only access to databases (Rymer, 1992, pp. 16).

(3) Future Prospects: The maintenance and expense of open gateways may soon be made unnecessary by the introduction of standard Application Program Interfaces (API, to be covered in next section) from major vendors (Rymer, 1992, pp. 14).

d. PC Front Ends with Database Application Program Interfaces

Application Program Interfaces (API) provide a consistent means of access for a variety of client-based application programs. API's are being developed and marketed for a wide variety of functions that include database access, user authentication, group scheduling, calendaring functions, and document management (Petrosky, 1993, pp. 104). A database access API is activated from within an application, and allows that application to communicate more directly with an RDBMS than under the other interoperability options. Figure 4 illustrates APIs in a client server network.

Database APIs standardize the previously proprietary ways applications would submit queries to multiple databases. The API consists of a standard set of call routines, residing on the client, that accept a user's SQL statement and then hand it off to a driver that's programmed to deal with the specific target database. A different driver would exist for every type of server-based database. Prior to sending the request out over the network, the driver performs the functions of mapping the query to the actual database, validating the query, and making any required changes to the SQL code

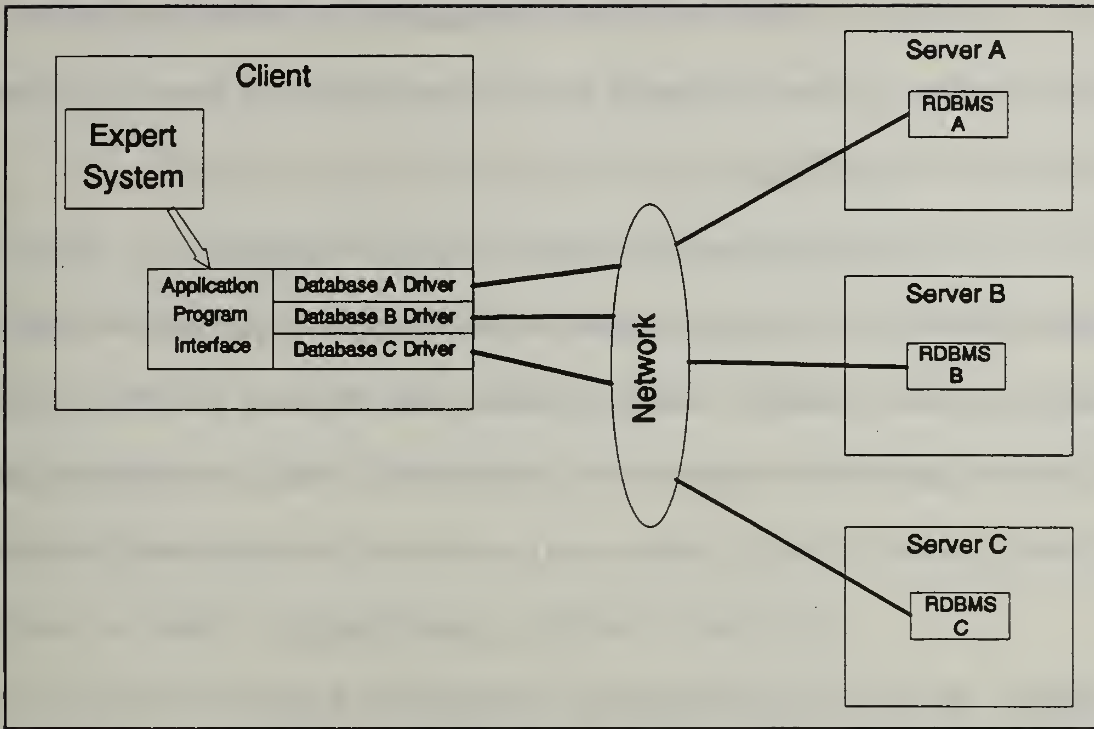

Figure 4: Application Program Interfaces

so that it may be understood by any unique features of the target database system (Rymer, 1992, pp. 9). When the results of the query return, the driver performs the same set of functions in reverse before handing the answer to the original application that submitted the query.

(1) Advantages: APIs allow software developers to create applications that access databases in standardized ways (by way of API calls) without having to re-invent such access within each user application.

API's provide access to a wide variety of server-based functions, of which databases are but one.

API's eliminate the need for some intervening layers of middleware, as in other options.

Competition among major vendors to produce API's is quite heavy. The information customer will benefit from this competition with lower prices and/or more feature-laden API's.

(2) Disadvantages: APIs don't yet encompass the means to communicate between client applications and multiple servers (Rymer, 1992, pp. 9). This leaves APIs limited to access of databases on the local network unless the organization has the technical know-how to intervene with a smart network that's capable of sending queries to the right database, and back, in a way that's transparent to the API.

APIs don't allow for a single query to operate on multiple databases. If such a query were required, it would have to be done as one query each to the multiple databases, and then the responses would be combined/enmeshed to consolidate the final answer within the client database.

(3) Future Prospects: API wars are likely to continue with each vendor trying harder to satisfy the market's needs for transparent multiple database access. Hopefully, the competing standards will eventually merge into a common set of API calls that can be used interchangably among applications and RDBMSs.

None of these APIs is yet poised to satisfy some of the potential high-performance requirements of expert systems or decision support systems. For example, post-processing, the aggregation of a set of queries PRIOR to returning the answer to the client application, is not doable in these solutions. Currently, a client

application must perform its own aggregation/refinement of data that's returned from a query.

e. Database Application Program Interface Alternatives

Competing vendors are working hard to establish their API as the accepted standard. By openly publishing their APIs, they compete for the attention of software vendors to use a particular API as part of their application software. Gaining wider acceptance of a given API is resulting in a competitive battle among three leaders for an emerging database API standard:

(1) SQL Access Group (SAG)

SAG is a consortium of database vendors who have defined a database API which uses ANSI SQL as its base. SAG specifies ISO's Remote Data Access (RDA), and TCP/IP as the network protocols that are required between clients and servers (Ricciuti, 1992, pp. 42). Forty-five vendors have signed-up to supporting the SAG API standard (as of Sep 92), and products are expected to become available sometime in 1993 (Ricciuti, 1992, pp. 39) (Johnson, 1992, pp. 30).

(2) Open Database Connectivity (ODBC)

ODBC is Microsoft's offering for a database API. ODBC uses the Named Pipes network interface, which is a part of the Microsoft LAN Manager protocol (Rymer, 1992, pp. 10). ODBC adheres to standard SQL format for queries submitted over the network to databases. Obviously, ODBC is a Microsoft offering that adheres to Microsoft developed standards, such as the Windows interface. With ODBC,

Microsoft is offering a set of functions that encompass those currently being offered by the leading server-based RDBMS products. If the RDBMS vendor offers an ODBC driver for their product (as Microsoft is encouraging them to do) then the client-resident driver maps calls from the ODBC API to its own set of functions. The query is routed to the DBMS and back in its own way, and the driver then reverses the process to pass the answer back to the ODBC API, and in turn to the original application (Finkelstein, 1993, pp. 48). With the right drivers, our expert system could access any RDBMS on its network via the ODBC API.

ODBC drivers are not yet widely available, but will be when Microsoft adds ODBC to its Windows graphical interface in a future release (Petrosky, 1993, pp. 104). ODBC has been implemented within Microsoft Access which is now on the market. Although APIs allow for an agreed upon method for interoperability, they do have a weakness of not allowing for some unique/proprietary functions in some RDBMS. In these cases, ODBC allows for a 'pass-through' facility which allows an application to send an RDBMS-specific call to the RDBMS (Finkelstein, 1993, pp. 49)

(3) Integrated Database API (IDAPI)

IDAPI is a standard still in development by Borland. Its name changed in Nov 92, and it was previously called the Open Database API (ODAPI) (Finkelstein, 1993, pp. 51). Like the SAG API standard, IDAPI will use the ISO Remote Data Access (RDA) network protocol. Borland promises a more robust API that's capable of submitting SQL queries to relational databases as well as record-oriented queries (i.e., non-SQL) to non-relational databases. The emphasis on record-oriented

queries allows IDAPI to communicate with dBase, which Borland owns, and dBase compatible products. Other major vendors who have joined Borland in this standard are IBM, Novell, and WordPerfect. Although IDAPI is yet to reach the market, its goals for database access are more ambitious than ODBC or SAG since it intends to reach non-relational databases, include non-SQL query languages, and allow for future introduction of object-oriented technology (Zuck, 1992, pp. 320).

E. Repositories

Repositories represent the future of database systems. They manage larger volumes of data than databases, and are the next evolutionary step in the series of ways data has been managed. A repository is a set of specialized information management facilities that manage databases (Jones, 1992, pp. 28). The concept of respositories is relatively new. As a result, it is often misunderstood and misnamed under a variety of vendor-attached labels and claims. IBM for example uses the term 'Information Warehouse' to describe their set of products that satisfy some concepts of a repository. Within standards groups, repositories are referred to as Information Resource Dictionary Systems (IRDS) (Jones, 1992, pp. 28). This section will explain repository theory, show a relation to the coming X.500 standard, and discuss its relevance to databases. Understanding repositories is essential to understanding the future of database interoperability.

1. Repository Theory

A repository views an organization's set of data as one entity and attempts to provide a cohesive means of identification and access for that information. Repositories manage a wider range of information than what we normally associate with databases. For example, it might encompass all databases, knowledge bases, document files, and images throughout an organization. A new range of services becomes available under repositories, all aimed at making more information accessible, sharable, and manageable. Goals of repositories include (Jones, 1992, pp. 30):

- To manage information that in turn manages information. A repository stores actual data, and data about that data (metadata). It can be viewed as a metadatabase that manages lower level data stores.
- To create views of data, regardless of how it's actually stored, that match the needs of users.
- It allows data to transparently appear to applications programs as a consistent useable set.
- It provides easy access to information, regardless of its original source.
- It allows information to be easily shared, within security constraints, both within and outside the organization.
- It provides the ability for applications to query multiple information sources transparently, and receive the answer as one consolidated response.

Repositories are planned to provide their services via a set of specialized facilities. These facilities would provide a layer of management over the various information stores within an organization. These facilities are (Jones, 1992, pp. 28):

- Reference Management Facilities - dictionaries, encyclopedias, thesauruses, glossaries.

- Directory Management Facilities - maintains data addresses and attributes for schemas.

- System Administration Facilities - manages the installation and maintenance of new information in the repository.

Establishing standards for repositories is a key issue because of the benefits that can accrue. If vendors market repository products that follow agreed upon standards, then not only will organizations gain more ability to manage information within their own boundaries, but that same information will become a sharable asset outside the boundaries of the organization. The X.500 standard is key to these benefits.

2. The X.500 Directory Services Standard

X.500 is the short name given by the Consultative Committee International Telegraph and Telephone (CCITT) to the standard for Open System Interconnection Directory Services. It makes standardized directory services available to applications so they can locate information about a database (Lawton, 1992, pp. 28). X.500 is the yet to be implemented standard that will form the basis for distributed database structures and respository systems.

In the terminology context of the previous section on database access, X.500 is technically an Application Program Interface (API) standard (Marshak, 1992, pp. 4). Its market acceptance as a standard may serve to standardize the competing vendor developed database API's into one all-purpose standard that simplifies database

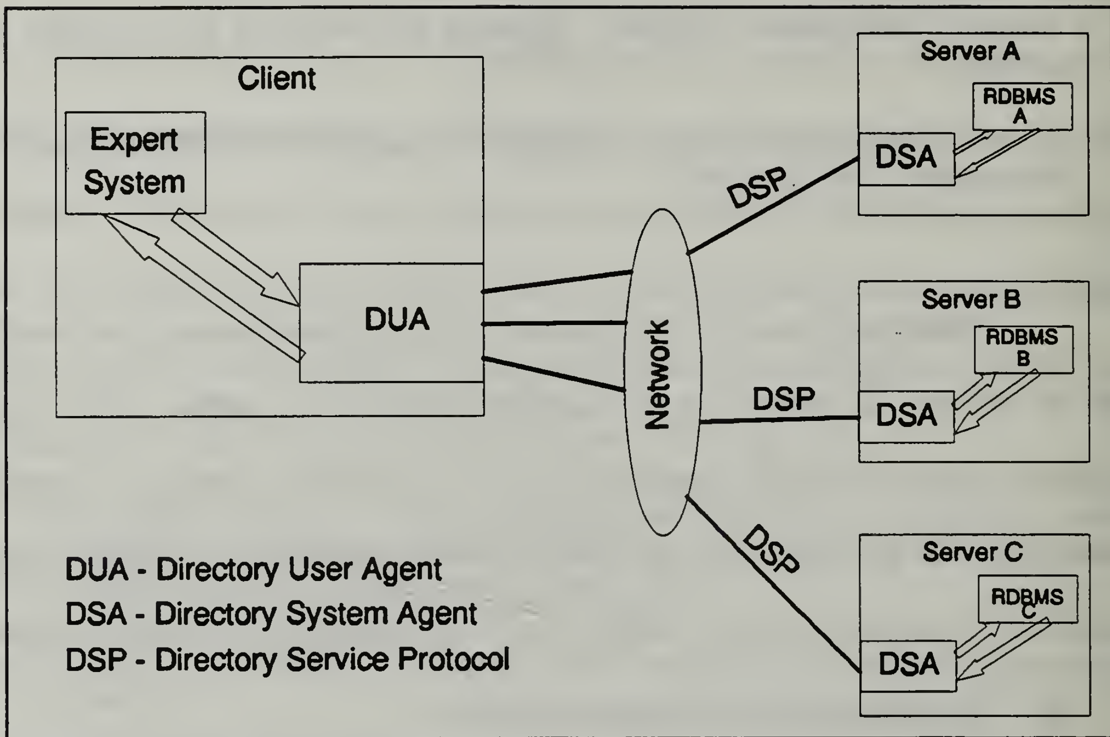

Figure 5: X.500 Query Process

interoperability. Figure 5 illustrates how X.500 is planned to work.

X.500 is implemented locally, at the server level, to provide a standardized directory of the information resident on that server. The DBMS that actually manages data on the server is separate from the X.500 directory module (Lawson, 1992, pp. 28). A client-based application submits queries via an X.500 Directory User Agent (DUA). Similar to an API, the DUA can be built into the application. The query passes to a Directory System Agent (DSA), which may satisfy the request directly, or pass it to the DSA who can. DSAs can work in sequence to allow a query to propagate to multiple databases, combining the answer into one concise report back to the original application

that requested it (Lawton, 1992, pp. 28). X.500 also encompasses the protocol used between DUAs and DSAs. This protocol is called the Directory Access Protocol (DAP) (Lawson, 1992, pp. 28).

3. Why are Repositories and X.500 Important?

Repositories, and the X.500 standard within them, have the potential to play a vital role in future information systems. Currently, the FBI and NASA are experimenting with X.500 directories that contain fingerprint images, mug shots, and photographs (Lawson, 1992, pp. 28). Large-scale repository implementations will dramatically increase the accessibility, timeliness, and value of information.

Current projections estimate that X.500 networks will begin to appear in 1994 (Miley, 1992, pp. 195). While it's likely that they will appear only in the largest organizations, the follow-on projection is they will be generally available in 1997. Although this technology will provide many benefits, it will come at the expense of more technically trained people, able to understand and implement systems that manage larger amounts of information. The skills that will be required of those people is the topic of the next chapter.

III. ORGANIZATIONAL IMPACT ON DATABASE MANAGEMENT FROM EXPERT SYSTEMS

A. OVERVIEW

The objective of this chapter is to discuss the organizational aspects of using expert system applications with relational databases. It focuses on the people skills that are required to successfully implement expert systems and relational databases. The chapter begins with a description of the standard jobs that exist within IS organizations to manage databases and expert systems. It then extrapolates into the future to anticipate the changes in those jobs that will take place as database and expert system technologies continue to evolve.

The skills that will be required of people who will manage future information systems are becoming a major concern to upper management within IS organizations. A recent survey of IS managers found that 'improving the IS human resource' and 'improving leadership skills in IS' ranked third and sixth, respectively, among their top ten concerns (McPartlin & Tate, 1992, pp. 82). As the potential gains to be made from databases and expert systems continue to grow, so too must the managerial and technical skills of the people who manage and maintain those systems continue to grow.

B. THE PEOPLE ROLE IN MANAGING RDBMSs

Professional positions dedicated full-time to data administration first began to appear in IS organizations in the early 1970's (Leong-Hong, 1982, pp. 207). At first, these people performed purely technical functions and were given responsibility for databases and DBMSs. Over time, their functions evolved to be both administrative and technical. The details of these functions will be described next, but the basic result was the evolution of the Data Administrator (DA) and the DataBase Administrator (DBA). The combined functions of the DA and DBA positions, and their staffs, fulfill the requirements to manage an organization's data resources. The people resources that are committed to these functions vary greatly from organization to organization (Leong-Hong, 1982, pp. 208). In a small IS organization, all these functions might be satisfied by one person. At the other extreme, in a large IS hierarchy, the DA and DBA functions might be separate offices, filled by relatively high-ranking people, each with his or her own staff. At either extreme, or somewhere in the middle, understanding the responsibilities of a data administrator and a database administrator sets the baseline for predicting the skills that will be required in the future.

1. Data Administrator

> A Data Administrator (DA) is: A person or group that ensures the utility of data used within an organization by defining data policies and standards, planning for the efficient use of data, coordinating data structures among organizational components, performing logical data base designs, and defining data security procedures (DoD Directive 8320.1, 1991, pp. 2-1).

As the name implies, a DA is primarily responsible for the administrative functions of managing an organization's data resources. As such, a DA relies on

managerial and administrative skills to gain a strategic view of information's value to her organization. This requires an ability to interact among groups within the organization and determine what data should be in the organization's databases. The DA is also responsible for establishing the organization's data policies and standards.

It is common for DAs to complain of not having enough authority. Successful data administration requires the DA to be visible, well-positioned and recognized throughout the organization. DAs can accomplish these goals by communicating to upper level managers the benefits of data administration and how a strategic data resource is an investment for the future. For all these reasons, it's important for a DA to have strong interpersonal skills.

With respect to expert systems, and other applications, the DA's policies define the interface between users, DBA's, and application programmers within the organization (DoD Directive 8320.1, 1991, pp. 3-2). These policies are important because they impose the discipline that enforces a strategic view of data within the organization. Without such discipline, application developers are prone to define data requirements on an application-by-application basis. This can result in a proliferation of smaller independent databases, each tied to one application, with increasing amounts of data redundancy and inefficiency. With enforcement of proper DA policy, a strategic data resource can be established, cultivated, and maintained for shared use by most, if not all, user applications.

DAs are responsible for defining a common information perspective for the organization. This is done by establishing a data dictionary which requires a DA to have

knowledge of the organization's data and the business rules that lie behind it (Halle & O'Neil, 1993, pp. 11). Data dictionaries are a component of most relational DBMSs and provide the basis for a DA to implement the organization's data policies. Once established, the maintenance of the data dictionary remains a DA responsibility.

DAs are also responsible for establishing and maintaining the organization's information model. This model provides the strategic design of information throughout the organization, and it helps to optimize the way data is stored based on the particular ways applications use the data and the transaction volumes that are expected. Data models cause a top down approach to data planning and design and result in a normalized database that can be shared by multiple applications, as opposed to individual application databases (Takoushian, 1992, pp. 58).

2. Database Administrators

> A DataBase Administrator (DBA) is the person responsible for the physical design and management of the database and for the evaluation, selection and implementation of the DBMS. In smaller organizations, the database administrator and data administrator are one in the same. However, when the two responsibilities are managed separately, the database administrator's function is more technical (Freedman, 1992).

As stated in the above definition, the DBA's functions start where the DA's functions stop, and tend to be more technical in nature. The DBA is the person who sets the DA's policies in action by using the DBMS's facilities to establish and optimize the normalized data tables that comprise the organization's database.

Database access, security, and integrity are some of the DBA's most important functions. The DBA insures authorized access to read and/or write to the

database by maintaining access controls on a user by user basis. These controls prevent the unauthorized access, copying, updating, or destruction of any part of the database (Leong-Hong & Plagman, 1982, pp. 211). Relational DBMS products provide the means to maintain access controls to data at varying levels of detail. For example, a DBA, based on the DA's access policy, may provide supervisors with read-only access to the salary information of those who work for them, while limiting write access to that same information only to certain individuals within the personnel department.

The DBA performs database operation, maintenance, and management functions that ensure the technical well being of the database environment (Leong-Hong & Plagman, 1982, pp. 211). Foremost within these responsibilities are establishing the backup, restart, and recovery procedures that ensure the database can be saved and restored despite a variety of disasters that may occur. The DBA also maintains current database definitions within the data dictionary as changes occur. He is also responsible for the configuration and installation of new versions of RDBMS software.

On a day to day basis, the DBA monitors the database environment and takes actions to keep database performance at a high level. Many RDBMS's include performance tools that can provide information on how well the database is performing. The DBA uses this information to monitor database activities, identify bottlenecks, and fine tune the database for optimal performance. Database tuning actions usually involve trade-off decisions that require a strong technical understanding of the DBMS, its interactions with numerous applications, and the hardware limitations of the computers and network in use.

Finally, the DBA must establish a liaison with a variety of people to maintain the database. First, he trains end-users on how to use the database. Second, he provides guidance to application programmers on how to make efficient use of the database within applications. Third, he consults with systems analysts to fine tune the DBMS hardware and software in concert with the operating systems (Leong-Hong & Plagman, 1982, pp. 213). Lastly, and most importantly, he interfaces with the DA so together they can provide for the consistent organizational use of data within the organization (DoD Directive 8320.1, 1991, pp. 2-1).

A typical DBA want-ad would request a minimum of three years in programming, systems analysis and database analysis. A knowledge of systems software and relational database experience would be required. Problem-solving ability and business experience would be a plus. A bachelors degree in computer science or information systems (IS) would be required (Goff, 1992, pp. 179).

3. Upper Management

Information systems literature is replete with references to the importance of top management to the success of database systems. The consistent message for top management is that their strong involvement and support is required to successfully implement database systems within their organizations. When strategic data planning is left to IS staff, without top management involvement, the result tends to suffer from a lack of business experience and the strategy becomes the basis for organizational political in-fighting (Martin, 1989, pp. 10).

There are two major benefits that result from top management support for strategic data planning (Martin, 1989, pp. 10). First, their support lends credibility to the effort in a way that forces cooperation from non-IS portions of the business, resulting in an accurate, supported, and understood data model. Second, the act of coming up with a strategic data plan, in and of itself, has been shown to help organizations gain a 'strategic vision' that helps them clearly understand where they are and where they are going (Martin, 1989, pp. 10).

C. THE PEOPLE ROLE IN MANAGING EXPERT SYSTEMS

1. The 'Expert'

> An Expert, also commonly referred to as the domain expert, is a person who has the special knowledge, judgement, experience, and methods, with the ability to apply these talents to give advice and solve problems. It is the domain expert's job to provide knowledge about how he or she performs the task that the knowledge system will perform (Turban, 1990, pp. 434).

Although he's not necessarily an IS person, the expert plays a vital role in the development of an expert system. His role is fairly straightforward as the source of expertise to be tapped by the knowledge engineer. In the development of an expert system, one or more experts may contribute to the knowledge base. Documented sources of information such as textbooks, regulations, policy and procedure manuals, or catalogs may also contribute to an expert system's development. In this way, documented sources can complement, or sometimes even replace, the expert. The experts who tend to work best are those who are knowledgeable, articulate, and have a reputation for finding good solutions to problems in the expert system domain (Waterman, 1986, pp. 9).

2. Knowledge Engineer

> A Knowledge Engineer (KE) is a person, usually with a background in computer science and artificial intelligence, who knows how to build an expert system. The KE interviews the experts, organizes the knowledge, decides how it should be represented in the expert system, and may help programmers write the code (Waterman, 1986, pp. 9).

As may be implied from the above definition, the KE is the most important person to the development of an expert system. In its simplest form, KEs interview experts in a particular domain of interest, and develop a program with rules that recreates the approach to the problem (Goff, 1992, pp. 91). A KE may work alone to develop small expert systems, or may lead an expert system development team for larger systems. Being a successful KE requires strong interpersonal communications skills, a knowledge of programming languages, and prior experience with expert systems and the software products that are used to develop them. Knowledge engineers must be skilled at eliciting large volumes of information from experts and documented sources, and then crafting that information into a knowledge base. Excellent interpersonal skills are required to successfully communicate with experts and illicit the right information on which to base the expert system. Developing an expert system is a complex process because it requires one to work in meticulous detail with experts in advanced areas of work (Goff, 1992, pp. 91). KEs also need experience in programming languages, especially those used in expert systems such as C, Lisp, or Prolog.

KEs use a ten phased process to develop expert systems (Turban, 1990, pp. 446). These ten phases encompass system analysis and planning, system design, knowledge acquisition, knowledge representation, and implementation. Throughout the

process, the KE is the person primarily responsible for development and implementation of the expert system.

D. NEW ROLES FOR COMBINED OPERATIONS

In the context of the information previously presented in this thesis, there are several factors at work that will change the roles of people who manage databases and expert systems. Some of these factors are:

- a growing requirement to share information between organizations electronically as distributed databases become commonplace.

- an increasing number of users submitting more transactions as repositories become more common, hold more kinds of information, and are able to satisfy more needs.

- systems with increasing technical complexity as expert systems access distributed databases, with all the middleware and network concerns that come in between.

- an increasing concern for database security as business requirements force the need for electronic access to people outside the organization.

These factors, and others, make it valuable to speculate on the effect these changes will have on the people who manage tomorrow's information systems. In a distributed database environment, where repositories and expert systems will become common, I have coined two titles for future IS jobs: Knowledge Administrator (KA) and KnowledgeBase Administrator (KBA). These titles emerge from the names of their current day 'predecessors,' the Data Administrator (DA) and DataBase Administrator (DBA), and are meant to reflect a merger of skills between the database and expert systems fields. This section will speculate on their activities and the skills that will be required, as well as those of upper management in their organizations.

1. Knowledge Administrator

The KA will inherit the DA's role in the organization and must have the skills to accommodate a more strategically important management role for the organization. The value of information will continue to grow in the future. As a result, the KA will play a critical role as a communications bridge between the organization's business-oriented executives and the technical support community (DoD Data Administration Strategic Plan, 1992, pp. 9). The KA's value to the organization will increase, but he will have to become more business oriented while at the same time remaining technically knowledgeable of what information systems can do. The KA will have an executive level range of skills and will be positioned within the organization as an equal to other high level executives.

The KA's role will no longer be limited to database management, but will expand into one of information resource management (Stodder, 1993, pp. 40). Repositories will become the responsibility of KA's. They will be expected to proactively recognize, understand and then communicate the business opportunities that will result from investments in information technology. The inclusion of external databases and public access databases will serve to make this function more challenging. When strategically viewed in retrospect, the organization's 'knowledge' will have become a commodity in much the same way that we view 'data' as a commodity in today's organizations. The organization's flexible ability to access external knowledge will also become a valuable commodity and will be a responsibility of the KA.

DoD will not be immune to these changes. The 1992 DoD Data Administration Strategic Plan devotes a full section to speculation on what data administration will be like in the year 2000 (Department of Defense, 1992, pp. 7-9). Although the term KA is not used, the plan does foresee an increased management role to be played that involves repositories, distributed databases (referred to as 'corporate databases'), decision support systems, and a focus on standards that might allow the flexibility to share information electronically among international coalitions (Department of Defense, 1992, pp. 7-9). A faster pace of business mergers will require information systems that can adapt quickly, in the same way that joint forces and international coalitions must have C3 systems that can share information while retaining the required security constraints.

The information policies that KAs establish will become more strategically important to their organizations than those policies that DAs established in the past. The data dictionaries and information models KAs create will have to incorporate distributed databases, repositories, and the needs of expert systems. The KA will also act as the data liaison to people and resources outside the organization. As public access and distributed databases become more common, these external responsibilities will grow in importance.

2. KnowledgeBase Administrator

The KBA will inherit the DBA's role in the organization. But unlike the KA, the KBA's role will become more technically oriented and will require a higher degree of technical skills than are required of DBAs today. An ability to remain current in

technology, and apply that technology correctly to future systems will become indispensable.

KBAs will be technically challenged to remain current amidst the various changes that will take place in database technology. Their systems must be able to accommodate the increased numbers of users that will result from shared information. The nature of users will also change since, in the future, expert system queries will have the same impact as increased numbers of human repository users. Once developed, the easy duplication of expert systems holds the potential to dramatically increase the 'user' demands on repository systems.

While KAs will become further integrated within the executive levels of the organization, KBAs will have to become more integrated with other technical positions within the organization. Distributed databases will force a closer relationship between KBAs and network technicians. Implementation of the 'middleware' described in chapter two will combine the efforts of KBAs, network managers, and systems analysts so information can be available to meet the needs of more users (Radding, 1993, pp.36).

Repository access, security, and integrity will pose new challenges as systems become more complex, the volume of information increases, and the number of users grows. In war, be it military or business, the ability to compromise or destroy the enemy's information will become a threat that cannot be allowed to happen.

All of these factors, taken together, impose a heavy burden on the performance of repository systems. KBAs will have no choice but to depend on more

sophisticated tools to optimize and secure repositories. Performing these functions manually will become increasingly difficult to accomplish.

a. Using Expert Systems to Manage Repositories

Expert systems are beginning to emerge as the tools that will provide solutions to the technical management of tomorrow's information systems. Expert systems can already perform many of the roles of today's DBA, and they can be expected to continue to play that role in the future (Eliot, 1993, pp. 9). As today's databases, and tomorrow's repositories become larger and more complex, better ways of managing them are required, and expert systems can provide these solutions.

Expert systems and databases can be combined in many ways. For example, you can (Eliot, 1993, pp. 9):

- Use expert systems to scan databases to glean particular insights.
- Use expert systems as front-ends to databases, allowing programmers to use a larger variety of database development languages.
- Use expert systems to automate the tasks of DBAs in tuning RDBMSs for optimal performance.

X-Tuner is an expert system that can help databases achieve optimal performance (Eliot, 1993, pp. 10). It was built using the Nexpert Object expert system shell and has been used as a prototype system to improve the performance of Oracle databases. X-Tuner uses syntactic transformation to improve database performance by using its rule base to anticipate how well an existing RDBMS will be able to react to a given query (Eliot, 1993, pp. 10). X-Tuner is installed to receive an SQL query prior to its arrival at the Oracle database, and when applicable, transforms the query into a

more optimal form before passing it on to the database. It compensates for poorly constructed queries that would unnecessarily consume database resources if submitted in their original form. In some cases, query response time was reduced from over 30 seconds to less than one second (Eliot, 1993, pp. 10) .

3. Upper Management

Upper management will continue to demand that information systems (IS) professionals gain improved business skills in addition to their technical skills. This demand will be especially felt by KAs as organizations demand cost justification for IS, and users require information systems that are more responsive to their needs (Davis, 1993, pp. 29). Upper management will also become more aware of the strategic importance that information systems play in business success. For this reason, KAs will move up in rank and importance within organizations, and will be in a better position to gain support for IS. However, KAs will be successful only if they can effectively communicate, in business terms, how technology improvements to IS can strategically improve the organization.

In a more in-direct way, upper management demands will increase on KBAs. The tools they use to manage information will become more complex, while demands on information systems will increase. Improved technical skills will be required to configure and implement off-the-shelf products to meet the organization's needs. A preview of these products is the subject of the next chapter.

IV. COMMERCIAL PRODUCTS - THEIR POTENTIAL FOR COMBINED USE

A. OVERVIEW

The objective of this chapter is to review a set of commercial products that perform the functions discussed previously in this thesis. The commercial products described in this chapter are intended to provide a representative sample, from among other comparable products, of what could be used to establish expert systems that interact with a relational database. The particular product choices are not intended as a competitive review or price ranking of products. Such rankings are readily available in computer journals, and a repetition of such a review here would soon become outdated in the competitively fast-paced world of computer software.

Instead, this chapter reviews a set a commercial products as a means of exposing the reader to one set of software products that could be chosen for an information system that supports expert systems interacting with relational databases. This look at commercial products also offers an opportunity to see the specific ways vendors implement the generic features outlined in Chapter II, as well as providing a glimpse of the sometimes 'flashy' and confusing terminology used to describe their features. The set of products are presented in the context of configurations as they were presented in Chapter II.

B. PRODUCT CATEGORIES

As was shown in Chapter II, using expert systems with databases can involve varying configurations of products based on the particular requirements and the organization's installed base of hardware, software, and communications networks. As generically illustrated in Figure 6, there are three general categories of software products that can be used within these combinations: expert systems, middleware, and RDBMSs.

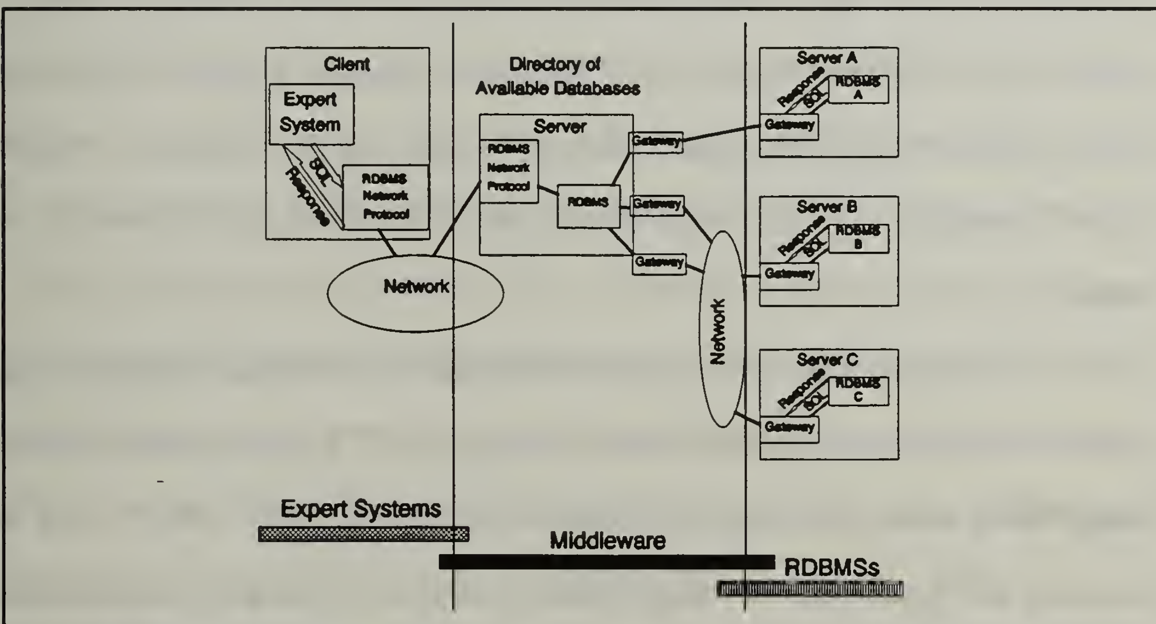

Figure 6: Product Categories

A particular expert system and relational database implementation may or may not require software from all three categories. The existing network structure, for example, may obviate the need for middleware. The overlap in product features between categories can also eliminate the need for purchases in all three areas. For example, an expert system product may include database application program interfaces (APIs) that obviate the need for middleware to perform that same function. Finally, within each of

the three categories, there is a wide spectrum of choices available. For example, within relational databases the spectrum ranges from low cost individual-use products for PCs, such as Paradox, all the way up to large-scale products such as Oracle or Sybase.

1. Expert Systems - Nexpert

Nexpert Object is an expert system shell developed and sold by Neuron Data Inc. of Palo Alto, CA (PC-Select, 1992). As an expert system shell, Nexpert provides the range of software tools needed to design, develop, implement, and maintain specific expert systems. Different Nexpert Object modules are available that allow the product to run on a wide variety of hardware, operating systems, and user interfaces. Nexpert Object is comparable to other expert systems shell products that are available on the market.

The initial stage of expert system development is knowledge acquisition from experts and documented sources (Turban, 1990, pp. 446). A Nexpert Object module called Nextra assists in knowledge acquisition (Neuron Data Inc., 1991). Prior to interviews with experts, the knowledge engineer uses Nextra to list and rank the entities and factors relevant to the expert system being designed. During the interviews, Nextra becomes an interactive tool that provides structure and helps focus on the important items of expertise. If multiple experts are interviewed, Nextra can track their inputs, identify conflicting points of view, and offer suggestions to help achieve consensus (Neuron Data Inc., 1991). When knowledge acquisition is complete, Nextra can automatically create rules for a 'first draft' prototype expert system.

Nexpert Object has its own graphical interface, or can be adapted to make use of previously installed text or graphical interfaces such as DOS, Windows, or Presentation Manager. Nexpert's interface is also used by programmers during expert system development, and has been found to improve productivity (Neuron Data Inc., 1991). Within an application, the interface would allow information to be presented in text, graphically, and/or in images.

Nexpert Object's set of programmable functions are provided as a programmer's library that can be individually called via an API (Neuron Data Inc., 1991). As a result, Nexpert Object code can be written as a stand alone expert system, or can be embedded within already existing applications that have been written in C, Cobol, or Fortran (Neuron Data Inc., 1991). This adds the option to embed modules of expert system intelligence within existing applications. Nexpert Object functions include the ability to query and process data from multiple different databases (Neuron Data Inc., 1991). Nexpert Object's database APIs allow direct access, for reading and writing to databases from Oracle, Rdb, Sybase, or Informix (Neuron Data Inc., 1991).

The Nexpert Object inference engine offers a variety of methods for knowledge processing. It is a rule-based system which can perform forward or backward chaining, or a mixture of the two, as its reasoning method (Stearns, 1992, pp. 12). Help and explanation facilities are available to ease the programming burden of adding such features to an expert system, and probability factors can be applied to the choices within a logic chain (Stearns, 1992, pp. 12).

Again, it's important to stress that Nexpert Object is representative of other similar expert system shell products that are on the market. Some of Neuron Data's competitors are the Aion Development System by AICorp., Mercury by Artificial Intelligence Technologies, and ProKappa by Intellicorp (Stearns, 1992, pp. 6).

2. Middleware - SequeLink

Middleware is the term that describes a growing market of software products that can be used to provide transparent access for client applications to server-based data. Middleware products are especially targeted to organizations trying to integrate client/server capabilities into existing information systems that include older components, such as mainframes. In such situations, older components in an information system can limit or prevent client applications from directly interacting with server databases to obtain data. Middleware products compensate for these limitations, and provide the means for client applications to gain access to data. SequeLink, by Techgnosis Inc. of Boca Raton, Florida, is the choice to represent middleware products.

SequeLink works by providing software modules that allow various client/server combinations of applications, operating systems, and networks to interact. There are five categories of SequeLink software modules:

- **Client Applications** - these modules are designed for use with specific applications such as Lotus 1-2-3, SmallTalk, Toolbook, and C language programs (Techgnosis Inc., 1993).

- **Client Operating Systems** - these modules are tailored to the client's operating system (DOS, Windows, OS/2, Unix...) (Techgnosis Inc., 1993).

- **Network Protocols** - these modules are specific to the network between the client and server, and can accommodate combinations of differing protocols over different networks (Techgnosis Inc., 1993).

- **Server Operating Systems** - these modules are tailored to the server's operating system (Unix, OS/2, MVS, VAX/VMS...) (Techgnosis Inc., 1993).

- **Relational Database Management Systems** - these modules are specific to the RDBMS in use (Oracle, Sybase, DB2, Informix, Ingres...) (Techgnosis Inc., 1993).

When installed, the SequeLink modules extend their associated software's functions to allow for transparent linkage between client applications and server databases (Techgnosis Inc., 1993). SequeLink functions are then embedded within commands in client applications. For example, SequeLink's Microsoft Excel spreadsheet module allows database query functions to be added within Excel command menus (Robertson, 1992). To execute these queries, the end user simply selects them as he would with any other Excel command.

3. Relational Database Management System - Sybase

Relational database management systems are the final category of products in our information system. In this category, a wide variety of products are available ranging from single-user PC-based products like Paradox, to large-scale server and mainframe based systems. Products at the larger end of the scale are designed to satisfy the needs of thousands of on-line users, and can provide the platform for customized strategic information systems such as airline reservation systems. Because of their large-scale strategic nature, database products such as Oracle, Sybase, or Ingress really consist of a family of products that can be configured to accommodate a wide range of corporate

information system needs. These products comprise a fiercely competitive market, where the players are constantly adding new features and improvements. The Sybase relational database system, by Sybase Inc., is the product chosen to represent this category.

Sybase is actually a family of products that can be configured to provide an advanced client/server environment. The Sybase family consists of four parts:

- Sybase Open Client
- Sybase Open Server
- Sybase Open Gateways
- Sybase Database Remote Procedure Calls (RPCs)

a. Sybase Open Client

Sybase Open Client is a set of software tools that allow programmers to develop applications able to access a variety of databases (Sybase Inc., 1993, pp. 2). As the name implies, these tools develop customized client-based applications, or can be used to add database access functions to existing applications (Sybase Inc., 1993, pp. 9). Structured Query Language (SQL) queries can be embedded within expert system applications using Open Client. Along with these development tools, Open Client includes a selection of application programming interfaces (APIs) that simplify connectivity to Sybase and non-Sybase databases.

b. Sybase Open Server

Sybase Open Server is a set of software tools that allow Sybase and/or non-Sybase databases, and other data sources to become open sources of information able

to support many simultaneous users (Sybase Inc., 1993, pp. 10). Open Server makes information available from servers, in response to requests from Sybase Open Clients, while maintaining control and insuring data integrity. Open Server can also be used to integrate non-traditional data sources into the set of information that's available to applications. For example, Open Server has been used to maintain on-line links to data residing within telephone switching systems, sensor networks, and stock quote systems (Sybase Inc., 1993, pp. 10).

c. Sybase Open Gateways

Sybase Open Gateways provide a means of application access to data residing in non-Sybase databases. These gateways can provide application access to Oracle, Rdb, Ingress, Informix, RMS, and DB2 databases (Sybase Inc., 1993, pp. 14). The gateway allows an application to query for data within a particular vendor's database in the native language of features of that database (Sybase Inc., 1993, pp. 14).

A separate product within the Open Gateway family is the Sybase OmniSQL Gateway. This product provides a single means of access to multiple, heterogeneous databases. It functions in much the same way as the conventional gateway described in Chapter II, and illustrated in Figure 2. When an SQL query arrives, the OmniSQL gateway uses its embedded catalog of attached databases to scan the request and route it to the appropriate database for processing. This product also allows distributed joins, which are SQL transactions that require the joining of data tables from separate databases (perhaps Oracle and DB2) in order to process the query (Sybase Inc., 1993, pp. 15). Finally, the OnmiSQL Gateway includes embedded optimizers that

review queries and determine the most efficient method to process requests that involve more than one database (Sybase Inc., 1993, pp. 15).

d. Sybase Database Remote Procedure Calls (RPCs)

RPCs are the last member of the Sybase family. They are a communications mechanism that allow client applications to efficiently request data from one or more server databases. Functions performed by Sybase RPCs are generically referred to as stored procedures. A stored procedure is a compiled set of code, residing on a server, waiting to be triggered by a call from a client-based application. Stored procedures are most valuable when they replace complex, often-used SQL queries. The Wal-Mart weather expert system referred to earlier provides a good example to illustrate. Figure 7 illustrates, in 6 steps, how a stored procedure simplifies this recurring process.

Lets assume that in the course of this expert system's consultation, an SQL query is sent out to retrieve weather data from a remote server on snow conditions in the northeast United States. This particular query is quite complex, and in-turn calls for the joining of database tables on two other remote servers to satisfy the request.

Rather than transmit a lengthy and complex SQL command, the client application transmits a call to execute the equivalent command, in its compiled stored procedure format, as it resides on the server (step 1). The server executes the stored procedure, which results in two SQL queries being sent to their respective databases (steps 2 & 3). The original server receives the data (steps 4 & 5), and according to the procedure, combines it into the pre-determined format for use by the client application.

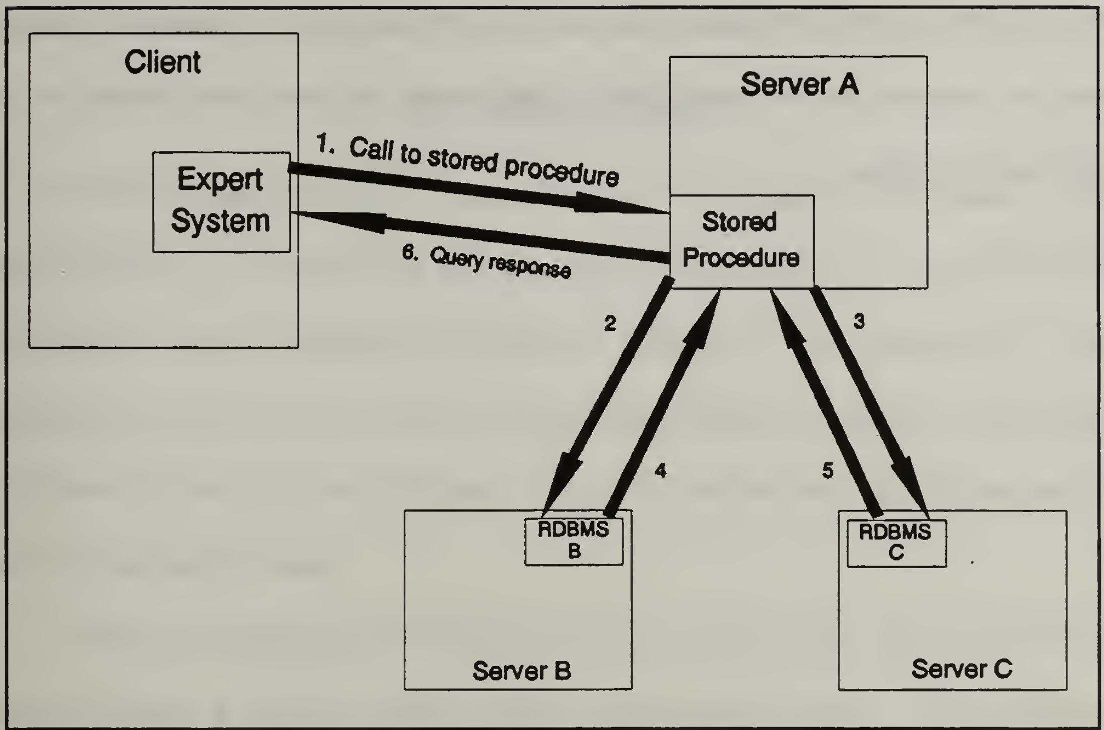

Figure 7: Stored Procedures

The query response is then sent to the original application (step 6), and the Wal-Mart expert system makes use of the data to determine appropriate snow shovel inventories for its stores in New Jersey.

e. Sybase Summary

Finally, it's important to stress that Sybase is representative of other relational database product families on the market today. Oracle, Ingress, and others have similar capabilities, each with its own unique set of terminology to make the product appear different and more advanced. The three categories of products covered in this chapter offer a bewildering array of choices that can easily confuse information system managers. The organization of this chapter is offered as a framework within which these choices will make more sense. When products are categorized, and then

viewed in the context of the distributed database alternatives from Chapter II, it becomes easier to compare the advantages and disadvantages that they may provide in your information systems.

V. CONCLUSION

This thesis has provided a management guide for future information systems strategic planning. It has focused on the potential benefits that can be gained from an integration of expert systems and relational databases. An integration of these components can offer powerful tools for knowledge management within an organization. The future information system challenges that face an organization in this area are both technical and people related.

Technical challenges result from decisions to be made over which hardware and software systems to choose, and how to best network them together. As is usually the case, organizations with existing information systems that have accumulated over the years can face even more complex decisions when trying to integrate new technology into older systems. As was shown in Chapter II, there are four general approaches that can be taken to integrate relational databases with expert systems. Also addressed in Chapter II were the concepts of application-independent design for databases and maintaining a loose coupling between applications and data. When followed, both these concepts allow for information systems that can grow and maintain the flexibility to adapt to future needs.

People related challenges stem from the increasing number of skills that are required to develop and maintain expert systems and relational databases. In the same way that database systems have evolved to require specialized groups of people to

perform development and maintenance, it's reasonable to expect a similar evolution will occur with expert systems. If integrated properly, I foresee a single set of positions for the development and maintenance of expert systems and databases. The term knowledge administrator was coined and described in Chapter III as the key member of this team.

Today's variety of software products offers a confusing array of choices to make in forming an integrated system of expert systems and relational databases. New offerings and updated versions of these products become available on a daily basis. Chapter IV offered a review of three products that span the categories of expert system shells, relational databases, and the middleware that integrates them.

Mr. Peter Drucker, the renowned management consultant, has reported that although the labor, materials, and energy required to manufacture a unit of output have each decreased at a compound rate of 1% a year since 1900, the amounts of information and knowledge required to manufacture a unit of output have increased at a compound rate of 1% a year (Drucker, 1992, pp. A10). These increases in knowledge and information began in the 1880's, coinciding with the invention of the telephone (Drucker, 1992, pp. A10). As more and better technology becomes available to handle information, one can only expect that the amounts of knowledge and information will grow at accelerating rates. In the same way that the bulldozer and the assembly line provided the tools to 'automate' the hand labor of millions of people, we are now seeing the emergence of tools that will improve the ways we will handle the ever-growing onslaught of information we will have to deal with in the information age.

LIST OF REFERENCES

American National Standards Institute. Database Language - Embedded SQL. Document ANSI X3.168-1989, New York, April 1989.

American National Standards Institute. Database Language - SQL with Integrity Enhancement. Document ANSI X3.135-1989, New York, October 1989.

Caldwell, Bruce. "The Answers Are Blowin' In The Wind." Information Week. 22 March 1993.

Celko, Joe. "SQL2: The Sequel to SQL." UNIX Review. December 1992.

Date, C.J. An Introduction to Database Systems. Vol 1, 5th ed., Addison-Wesley Publishing Co. 1991.

Davis, Dwight B. "Hard demand for soft skills." Datamation. 15 January 1993.

Department of Defense. Data Administration Strategic Plan. Defense Information Systems Agency, Center for Information Management. 14 July 1992.

Department of Defense. Directive 8320.1, DoD Data Administration. 26 September 1991.

Drucker, Peter F. "Planning for Uncertainty." The Wall Street Journal. 22 July 1992.

Eliot, Lance B. "Tuning a Database Expertly." A. I. Expert. January 1993

Finkelstein, Richard. "Making Connections Across the Enterprise: Client-Server Middleware." DBMS. January 1993.

Freedman, Alan. The Electronic Computer Glossary. The Computer Language Co. Inc. 1992.

Goff, Leslie. "Database Administrators: Super Stress Handlers." Computerworld. 16 November 1992.

Goff, Leslie. "Knowledge Engineers Blend People Skills, Programming." Computerworld. 13 April 1992.

Halle, Barbara and O'Neil, Bonnie K. "The great debate: Where do business rules fit in a client/server world?" Database Programming & Design. February 1993.

Hansen, Gary W. and Hansen, James V. Database Management and Design. Englewood Cliffs, N.J.: Prentice Hall.

Johnson, Amy H. "ANSI Out of Touch." Corporate Computing. September 1992.

Jones, Mark R. "Unveiling Repository Technology." Database Programming & Design. April 1992.

Keyes, Jessica. "Knowledge Systems and Case Converging." Software Magazine. September 1992.

Korzeniowski, Paul. "SQL Interoperability Still Hard to Find." CommunicationsWeek. 31 August 1992.

Lawton, George. "Multiplatform Directory Services: Will Future Networks Dial X.500?" MacWEEK. 28 September 1992.

Leong-Hong, Belkis W. and Plagman, Bernard K. Data Dictionary/Directory Systems - Administration, Implementation, and Usage. New York: John Wiley & Sons, 1982.

Marshak, David S. "The Quest for Common Mail APIs: Clearing Up the Confusion." Patricia Seybold's Office Computing Report. August 1992.

Martin, James. Strategic Information Planning Methodologies. Englewood Cliffs, NJ: Prentice Hall, 1989.

McPartlin, John P., Tate, Paul. "The skills to pay the bills" Information Week. 8 June 1992.

Miley, Michael. "Directory Assistance: What Can You Do Until X.500." Corporate Computing. November 1992.

Neuron Data Inc. Nexpert Object Version 2.0B Brouchure. 1991.

Peters, Tom. Liberation Management, Necessary Disorganization for the Nanosecond Nineties. New York: Alfred A. Knopf Inc., 1992.

Petrosky, Mary. "The API Wars Begin." LAN Technology. February 1993.

Radding, Alan. "Five Types of Tools to Get Data From Legacy Systems." Software Magazine. 15 January 1993.

Ricciuti, Mike "Terabytes of Data - How To Get At Them?" Datamation, 1 August 1992.

Robertson, Bruce. "SequeLink Delivers Mac to SQL Server Connectivity." Network Computing. CMP Publications Inc., June 1992.

Rymer, John R. "Database Interoperability, A Comprehensive Approach to Database Access for the 1990's." Distributed Computing Monitor, Vol. 7, No. 10 (October 1992).

Salemi, Joe. Guide to Client/Server Databases. Ziff-Davis Press, Emeryville, CA, 1993.

Seybold, Patricia. "The ISO/ANSI SQL Standard: Where Are We?" Patricia Seybold's Unix in the Office. November 1991.

Stearns, Martha. Market Survey of Knowledge-based Expert Systems. AT&T contract deliverable, February, 1992.

Stodder, David. "Uncommon voices: Four Database Industry Leaders Discuss the Outlook for the Industry." Database Programming & Design. January 1993.

Sybase Inc.- Sybase Open Interoperability Brouchure. 1993.

Takoushian, Robert, "Guerilla tactics of the stealth DA (goals and objectives of data administrators)" Database Programming & Design. April 1992.

Techgnosis Inc. SequeLink Client/Server Middleware Brouchure. 1993.

Toffler, Alvin. Powershift: Knowledge, Wealth and Violence at the Edge of the 21st Century. New York: Bantam Books, 1990.

Turban, Efraim. Decision Support and Expert Systems: Management Support Systems. New York: Macmillan Publishing Co., 1990.

Viescas, John. Quick Reference Guide to SQL. Redmond, WA: Microsoft Press, 1989.

PC-Select. Data Sources Report. Ziff-Davis Publishing Co. 1992

Zuck, Jonathan. "Transparent Data Access: Cross-Platform, Cross-Format." PC Magazine. 29 September 1992.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria VA 22304-6145	2
2.	Library, Code 052 Naval Postgraduate School Monterey CA 93943-5002	2
3.	Director for Command, Control and Communications Systems, Joint Staff Washington DC 20318-6000	1
4.	C3 Academic Group, Code CC Naval Postgraduate School Monterey CA 93943-5000	1
5.	AFIT/NR Wright-Patterson AFB OH 45433-6583	1
6.	AFIT/CIRK Wright-Patterson AFB OH 45433-6583	1
7.	Dan C. Boger, Code AS/BO Naval Postgraduate School Monterey CA 93943-5002	1
8.	Will Short, CDR, USN, Code AS/SH Naval Postgraduate School Monterey CA 93943-5002	1
9.	Warren W. Snow, Maj, USAF 1921 Balzac Dr. Las Vegas NV 89115	1